An Open-Source Limit-Order-Book Exchange for Teaching and Research

Dave Cliff
Department of Computer Science
University of Bristol
Bristol, BS8 1UB, UK
csdtc@bristol.ac.uk

Presented at the IEEE Symposium on Computational Intelligence for Financial Engineering (CIFEr); Bengaluru, India, Nov20-21 2018. This PDF archive version incorporates minor corrections and amendments to the version in the CIFEr Proceedings.

Abstract*— Many of the world's major financial markets are *electronic*, in the sense that all communication among traders and internal record-keeping at exchanges is entirely mediated and executed by digital computer systems and associated communications networks; and many such markets are also highly *automated*, in the sense that they are heavily populated by automatic *algorithmic trading* system which have largely replaced human traders at the point of execution in many spot markets. This has created significant demand for people skilled in writing and managing algorithmic trading systems. To provide a complete education and training in this field it is highly desirable to allow students/trainees to study the operation of their own algorithmic trading systems running live on a real financial exchange, interacting dynamically with other automated traders. This paper describes the *Bristol Stock Exchange* (BSE), a simulator designed and developed to meet that need. BSE provides a full implementation of the Limit Order Book (LOB) at the heart of modern financial exchanges, and includes reference implementations of several well-known leading algorithmic trading systems. BSE allows users to submit a variety of order-types including market, limit, fill-or-kill, time-to-live, immediate-or-cancel, iceberg; orders for specific actions at market-open and market-close; and linked pairs of contingent orders. BSE can be configured to allow empirical studies of issues in order routing between multiple exchanges and the performance of cross-market arbitrage trading algorithms. BSE also has provision for varying the exchange's fee structure, including implementing maker-taker and taker-maker pricing models, The *Python* source-code for BSE, which has been under ongoing development and extension since 2012, along with extensive documentation, is freely available on the *GitHub* online public repository, and can be used as a public-domain platform for teaching and research.

Keywords*— *Simulation for education & training; Financial Engineering; Financial Markets; Trading Technology; Algorithmic Trading; Limit Order Book; Market Simulator.

I. Introduction

Present-day global financial markets are heavily populated by automated *algorithmic trading* systems, computerized systems largely replacing human traders at the point of execution. This is true now for all major markets around the world, whether for equities, currencies, commodities, fixed-income debt contracts, and for derivative contracts on all these classes of tradeable asset. Algorithmic trading systems, often referred to as "robot traders" or simply as "algos" (see e.g. [1]) perform the roles that were traditionally done by highly-paid human traders, but are capable of analyzing huge quantities of data and reacting in tiny fractions of the time required by human traders.

There is significant demand for people skilled in the art of writing algorithmic trading systems, who typically need to be not only highly numerate but also experienced in devising algorithms than can execute quickly and reliably: in highly automated markets, a speed advantage can be crucial, and this has motivated the development of high-frequency trading (HFT) systems which are refined to the nearest millisecond of execution time and can enter into and out of transactions on sub-second timescales. The rise of HFT in modern financial markets has proven to be a contentious issue (see, e.g. [2, 3, 24, 22, 26]) and has also been the subject of detailed academic studies indicating the dynamics of current markets may be significantly different from the dynamics of markets in times past, when most or all traders in the markets were humans (see e.g. [19, 7]).

Regardless of the controversy over HFT, there is a heavy demand among investment banks and investment fund-management companies for algo-trading "talent", i.e. for employees who are skilled in the design, implementation, and ongoing refinement of algo-trading systems. At the University of Bristol, a leading UK university, graduates from our computer science undergraduate and postgraduate degrees are keenly sought for such roles, and our students are very keen to learn appropriate skills and knowledge. This presents a challenge to us as educators: to provide a complete education in this field it is highly desirable to allow our students to learn by studying the operation of their own algorithmic trading systems running live on a real financial exchange, interacting with other automated traders. In an ideal world, students would be provided with the means (both technical and financial) to launch their own algo traders into real financial markets and to observe them trading "live", but this is simply not practicable: in addition to the obvious concern that a badly-programmed algo could lose a lot of money very fast; there are regulatory obstacles.

To address this need, I have developed a simple, minimal, yet accurate implementation of a financial exchange, designed specifically for teaching university-level students about how contemporary exchanges work and also to offer a platform on which the behavior of our students' own algo trading systems could be realistically evaluated. As is explained later in this paper, this type of simulator offers experiences that cannot be gained from simply working with time-series of historical trade data. The simulation platform is named the *Bristol Stock Exchange* (BSE). BSE offers an accurate simulation of the *Limit Order Book* (LOB), the core technology at the heart of modern financial exchanges, explained in detail below, and includes illustrative examples of several well-known algo trading strategies. BSE has been constructed to be easy for students to understand, and easy for students to extend by adding their own algo-trader program code. The motivation for constructing BSE came from earlier experiences with other more complex market simulations that had been constructed and used by postdocs and PhD students in my research group such as *Open Exchange* [11, 12] and *Exchange Portal* [29, 30]. BSE has been in development and ongoing use since 2012, when its source-code and extensive documentation were first made available as free-to-use open-source on the *GitHub* public repository. This paper describes the core concepts in BSE, its design and implementation, and gives case-studies of its use in masters-level Computer Science teaching and research at University of Bristol.

II. The BSE Limit Order Book (LOB)

The LOB is at the heart of many contemporary electronic exchanges including major national exchanges such as NYSE or NASDAQ in the USA, LSE in the UK, and other comparable exchanges around the world. In real exchanges, independent LOBs will be maintained for each of hundreds or thousands of tradeable assets. In the spirit of simplifying minimalism, BSE can be configured to offer only a single LOB, for a sole anonymous tradeable asset.

The LOB is a record, a data-structure, that updates in real-time. Changes in the LOB occur as traders in the market issue orders (also known as *quotes)* to the exchanges. Fundamentally, quotes are either *asks* (a.k.a. *offers*), i.e. orders to sell; or *bids*, i.e. orders to buy. Orders need to specify the trader's desired *price* per unit, and the number of units (the *quantity*, also referred to as *size*, or *volume*) that the trader wishes to transact. Orders may specify that the trader is willing to take the best available price currently available, known as a *market order*, or may instead specify a *limit price*: the maximum bid-price at which the trader is prepared to buy, or the minimum ask-price for a sale. Because the limit price specified in a trader's quote may be some way distant from the prices at which transactions are currently occurring, quotes are added to the LOB and aggregated together to give an indication of current levels of supply and demand in the market over a range of possible prices. Discussion of the order types available in BSE is given in Section III, below.

The LOB is divided into the *bid side* and the *ask side*. Each side shows a table of quote-prices and the total quantity available at that price, i.e. aggregated over all orders at that price. Both sides of the LOB are arranged top-to-bottom ordered best-to-worst, which means that the bid side is sorted in descending order of price while the ask side is sorted in ascending order of price. The difference between the best bid and the best ask is known as the *bid-ask spread*, usually referred to simply as the *spread*. The arithmetic mean of the best bid and ask, the mid-point of the spread, is known as the *midprice* and is commonly used as single-value indicator of current/likely market price, although as is pointed out in [6, Ch.1], a more informative measure known as the *microprice* also takes account of the quantities available at the best bid and ask prices: BSE re-computes the microprice after each update to the LOB. BSE traders can signal that they wish to accept the LOB's current best bid or best ask by issuing a market order, or by issuing any other type of quote that *crosses the spread,* i.e. to quote an ask priced at less than the current best bid (referred to as *hitting* the bid), or to quote a bid priced at more than the current best ask (referred to as *lifting* the ask). BSE, like real-world exchanges, allows traders to cancel any specific quote, and for the exchange to charge a cancellation fee. Like real-world exchanges, BSE writes a sequential record of time-stamped market events to a list data-structure referred to as the market's *tape*, which is conventionally shown in most-recent-first order.

Figure 1 shows an illustrative sequence of six frames as the LOB is updated to reflect a sequence of incoming orders. Each frame has LOB data on the left-hand side (LHS) and a graph on the right-hand side (RHS). The LHS of each frame shows at upper left, in red text, the time, prefaced with "T:"; immediately below that, in green text, is the current microprice which displays as "M: -- --" if there is currently not enough data on the LOB to compute the microprice. At LHS upper center, in yellow, are details of the most recently processed quote. Below that, at center LHS, is the LOB: a table showing the current array of bid and ask prices, with corresponding quantities: bid-side on the right (quantities followed by prices); ask-side on the right (prices followed by quantities). Below the LOB, at LHS bottom, the two most recent events on BSE's tape are displayed. The RHS graph in each frame shows the market supply and demand curves derived from the LOB at that instant. As is conventional, these graphs show price along the vertical axis and quantity along the horizontal axis. The supply curve, a dashed blue line, steps upwards while the demand curve, a solid red line, steps downwards. Unlike the simplifications shown in academic textbooks on microeconomics, the supply and demand curves are manifestly not smooth curves or straight lines, and the equilibrium point *P* (where the supply and demand curves meet) never appears on the graphs because, if ever it did, that would indicate that a trader had issued a quote that crosses or touches the spread, and a transaction would then instantly occur, eliminating *P* from the graph. Instead, the green cross on the vertical axis of the RHS graph is the current microprice. The last two frames in Figure 1(at 00:27 and 00:30) result in a transaction because the incoming quote crosses the spread: the transaction price, quantity, and time are written to the tape for each transaction that occurs.

III. Order Types in BSE

Traders interacting with the BSE LOB can submit a variety of types of order, explained in more detail below. In the simplest use-case, all traders in BSE can be restricted to issuing only limit orders: if the price of a limit order does not cross the spread then it is added to the appropriate side of the LOB, adding liquidity to the market (i.e., being a "liquidity maker"). However if it does cross the spread then it is treated as if it was instead a market order, immediately consuming (hitting or lifting) orders at the top of the counterparty side of the LOB, thereby being a "liquidity taker". This minimal usage is sufficient for replicating the various well-known prior trading-agent studies discussed in Section VI.

The additional order-types in BSE are common in most financial markets. When there are multiple orders all to be processed, orders of the same type are sorted by arrival time (i.e., the time the order was issued to the exchange) and are dealt with in age-order, first-come-first-served. When pending orders are of different types, they are dealt with in priority-sequence depending on their type. The BSE order-types are described here in order of decreasing priority. The order types currently implemented in BSE, and their associated three-letter acronyms are: market orders (MKT); limit orders (LIM); good-for-day (GFD); fill-or-kill (FOK); all-or-nothing (AON); immediate-or-cancel (IOC); and iceberg orders (ICE). There are also four special types of order that are operational only at the start and end of a trading session, i.e. at market-open and at market-close. These are: market-on-open (MOO); market-on-close (MOC); limit-on-open (LOO); and limit-on-close (LOC). Finally, linked pairs of orders can be submitted with conditions specifying what should be done with the second order, depending on what happens to the first order: these are known as one-cancels-other (OCO) and one-sends-other (OSO) orders. The order-types are explained in sequence below.

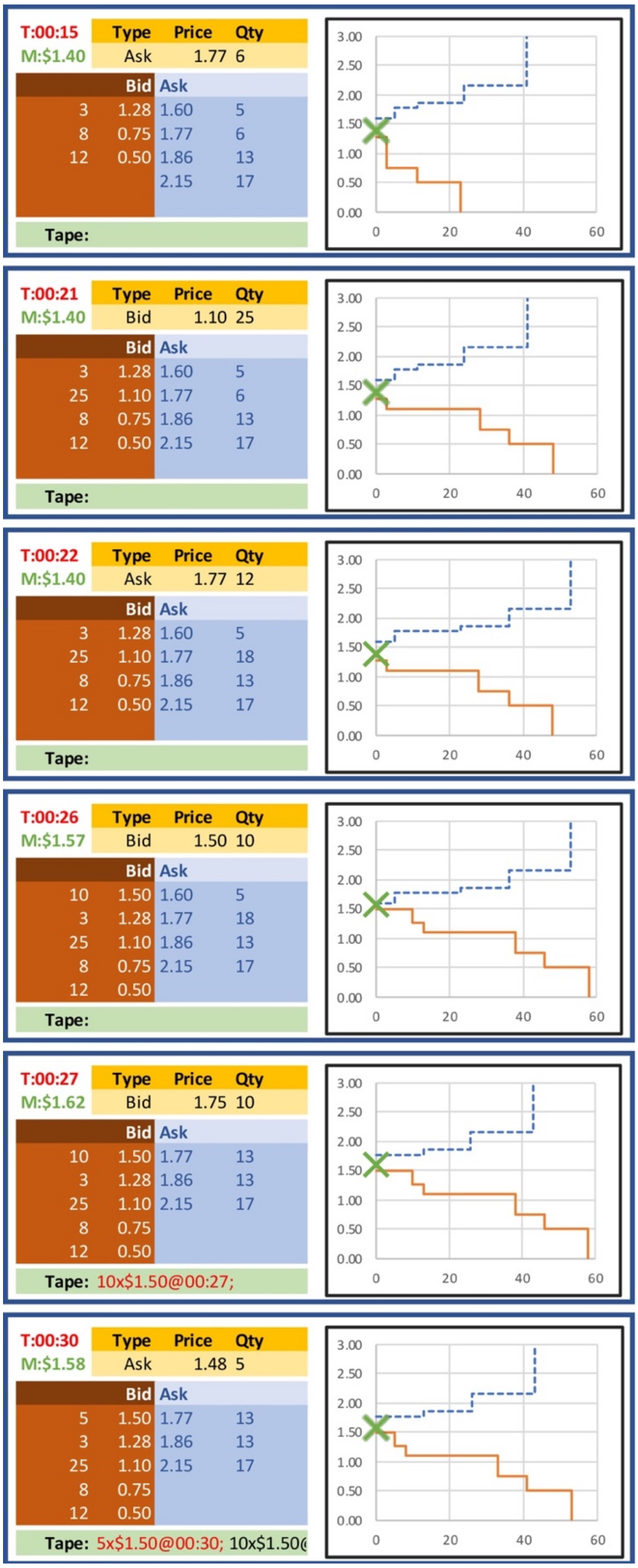

Figure 1: Six frames from an illustrative series of updates to the BSE LOB, from time=00:15 to time=00:30. Trades occur at 00:27 and 00:30, because the incoming quote crosses the spread: the most recent trade is written to the front of the tape. See text for further explanation and discussion.

In the descriptions that follow, p denotes the price specified in the order and q denotes the quantity (also known referred to as the *size* or the *volume*) specified in the order. The total quantity available on the counterparty side of the LOB (i.e. bid-side for an ask order, or the ask-side for a bid order) at prices better than or equal to p is denoted by Q_{PRICE}, and Q_{TOTAL} denotes the total quantity on the counterparty side of the LOB over all available prices.

A. *Market Order (MKT)*

MKT orders for quantity q execute at the current best available price, whatever that is, taking the best orders from the top of the relevant side of the LOB: a Bid-MKT lifts the q best-priced units on the ask side of the book; an Ask-MKT hits the q best-priced units on the bid-side. If $q > Q_{TOTAL}$, the unfulfilled portion of the MKT order, of quantity $q–Q_{TOTAL}$, is ignored. In BSE any limit orders with a price that cross the spread are executed as MKT. Variations on MKT, constrained by price p and specifying how the order is processed when $q>Q_{PRICE}$, are available in the other BSE liquidity-taking order-types FOK, AON, and IOC.

B. *Limit Order (LIM)*

LIM orders are added to the LOB in the manner described previously: if the LIM order's price crosses the spread, it executes as MKT; if not, it is added to the LOB

C. *Good-for-Day Order (GFD)*

GFD orders are time-limited LIMs: they are deleted from the LOB at the end of the trading "day", i.e. when the market session closes and accepts no more orders.

D. *Fill-or-Kill Order (FOK)*

A FOK order only executes if it can be immediately filled in full, consuming q units from the orders currently available on the counterparty side of the LOB at prices better than or equal to the price p of the FOK order: if that is not possible, the order is ignored: FOK orders do not partially fill.

E. *All-or-Nothing Order (AON)*

AON orders are similar to FOK orders. Whereas a FOK order specifies an immediate transaction, an AON order will sit on the exchange for a specified duration, waiting for the situation to arise where there is enough volume on the other side of the book for the AON to execute; if at the end of the duration the AON has not executed, the order is ignored.

F. *Immediate-or-Cancel Order (IOC)*

IOC orders differ from FOK orders by allowing partial fulfillment: if an IOC order of quantity q cannot be filled in full at prices at least as good as p because the LOB has $Q_{PRICE}<q$, the transaction goes through at quantity Q_{PRICE} and the remaining order at quantity ($q–Q_{PRICE}$) is ignored.

G. *Iceberg Order (ICE)*

ICE orders specify what maximum quantity q_{disp} for the order is displayed on the LOB: an ICE order of size q (typically $q>>q_{disp}$) is automatically split into a sequence of LIM sub-orders of size q_{disp}. As each sub-order is filled, the total remaining order-size q_{rem} is decremented by q_{disp}, and for each successive sub-order if $q_{rem}>=q_{disp}$ then another sub-order is issued, again for size q_{disp}; and when $0<=q_{rem}<q_{disp}$ the final LIM sub-order is issued with size q_{rem}.

H. *Limit-on-Close Order (LOC)*

When the current market session closes, if the limit price specified on a LOC bid order is equal to or greater than the closing best bid price, then the LOC bid executes; similarly for a LOC ask order, if the limit price is equal to or less than the closing best ask price, the LOC ask executes. If a LOC does not execute on market-close, it is automatically cancelled. LOC orders do not show on the LOB before close.

I. *Market-on-Close Order (MOC)*

MOC orders are MKT orders set to trigger immediately as the market trading session closes.

J. *Limit-on-Open Order (LOO)*

LOO orders can be issued in the *current* trading session, and rest at the exchange until the instant at which the market opens for the *next* trading session, when they are immediately issued as LIM orders, and processed appropriately.

K. *Market-on-Open Order (MOO)*

MOO orders can be issued in the *current* trading session, and rest at the exchange until the instant at which the market opens for the *next* trading session, when they are immediately processed as MKT orders.

L. *One-Cancels-Other Order (OCO)*

An OCO order involves a linked pair of constituent orders O_A and O_B: both are processed by the exchange and the LOB updated accordingly, in such a way that O_A executes then O_B is immediately cancelled, and vice versa. If both O_A and O_B could execute then only O_A does so.

M. *One-Sends-Other Order (OSO)*

OSO orders also involve a linked pair of orders O_A and O_B: O_A is processed first and, when it executes, O_B is then immediately sent for processing/execution. If O_A doesn't execute, then nor does O_B.

N. *Cancel Order (CAN) and Exit Market (XXX).*

Traders in BSE can cancel earlier orders that are still live on BSE, i.e. still awaiting matching with a counterparty. Each time a trader issues an order to the exchange, BSE responds with a unique order-identification code (OIC) for that order; so long as the order has not yet executed, a trader can subsequently send a CAN order, along with the OIC of the order to be cancelled: this removes that specific order from the BSE LOB and associated records. Each CAN order is written to the BSE tape. A trader can also send BSE an XXX order, which instructs BSE to cancel *all* of that trader's outstanding orders from the exchange's records. BSE implements this by issuing a sequence of individual CAN orders, and it is these that are written to the tape (that is, the single XXX order is *not* written to the tape).

IV. BSE EXCHANGE FEES

Real-world financial exchanges typically charge a range of fees for accepting and processing an order, with the precise amount charged depending on the nature of the order. Even cancellations of prior orders incur a fee on some exchanges. Devising trading strategies that are profitable in the face of such exchange transaction costs is an important aspect of working as a designer of automated trading systems. Historically it was common for the transaction fee to be calculated as a percentage of the order's total value (i.e., the per-unit transaction-price multiplied by the size/quantity of the order), with a "bulk discount" so that the percentage fee falls as the size of the order increases. In recent years, major exchanges have introduced so-called *maker-taker* fee structures where "maker" traders who provide liquidity to the market by posting limit orders away from the best price are given preferential treatment, in the form of reduced fees or even payments from the exchange; while "taker" traders viewed as removing liquidity from the market by hitting the bid or lifting the ask pay a higher fee than the makers: see e.g. [6, Ch.1] for further discussion. BSE is written in such a way that maker-taker (or the converse, taker-maker) fee/rebate systems can easily be implemented and explored.

V. MULTIPLE EXCHANGES

Although initially intended to offer the functionality associated with an individual financial exchange, BSE has been architected and implemented in such a way, using object-oriented *Python*, that it is trivially easy to create multiple exchanges all operating simultaneously. Having multiple exchanges in simultaneous operation offers opportunities for studying important aspects of contemporary financial markets such as smart order routing to ensure orders are executed at the best price available across multiple trading venues; and cross-market arbitrage where arbitrageur traders exploit discrepancies in prices between two or more trading venues, and/or make profitable use of differences in communications latencies between multiple markets (see e.g. [14, 37]).

VI. ALGORITHMIC "ROBOT TRADERS" IN BSE

The class of algorithmic "robot" traders implemented in BSE are known technically as *automated execution systems.* That is, the robot trader is assigned an *order* to *execute*, with the intention of maximizing *margin* on that order. This is a role that was previously performed by humans, known as *sales traders*. Orders come in from an external source and are commonly referred to as *client orders*. To better distinguish between a client order that a trader has been assigned to execute, and any orders that the trader sends to an exchange, client orders are referred to here as *assignments.* Assignments are typically one of the types of order defined previously in Section III. Market orders and any spread-crossing limit orders execute immediately, but other orders are added to the BSE internal records, typically showing on the LOB as quotes from the trader attempting to execute its assignment. The opportunity for a trader to make a profit comes if that trader can arrange a deal at less than the *limit-price* specified by the client order: so if the client asks to buy 1000 shares at a limit price of no more than \$140, the trader has a profit opportunity if he or she or it can instead secure a transaction at a price of \$138, saving \$2000; similarly when executing a client order to sell 1000 shares at a limit price of no less than \$150, if the trader can secure a deal at \$155 there is a potential profit of \$5000. The difference between the limit price and the transaction price is the trader's *margin* on that deal, usually expressed as a percentage of the limit price. Human sales traders would typically be working multiple client orders at any one time, and by acting as an intermediary they provide an anonymization service too. In recent years, human sales traders in the global financial markets have been almost entirely replaced by automated execution systems: "robot traders" or "algos".

This transition to automated algo trading systems has been enabled by roughly 25 years of research into what it takes to create a machine that can trade as well as a human, or better. That field of research has produced a small number of algorithmic trading strategies (originated by different authors, some working in microeconomics, others working in artificial intelligence and machine learning) which now form something of a *de facto* set of commonly-used benchmark algorithms, used for example when evaluating new trading strategies. The BSE *GitHub* repository includes source-code for the following trading algorithms, and their associated trader-code within BSE (an acronym or contraction of up to four characters) listed here in chronological order of first being reported in the literature: *Sniper* (SNPR)**:** inspired by Kaplan's prize-winning trading algorithm as described in [27]; *Zero Intelligence Constrained* (ZIC) introduced by Gode & Sunder [16]**;** *Zero Intelligence Plus* (ZIP: [8]); and Vytelingum's *Adaptive Aggressive* (AA) [35, 36]. The AA algorithm was later demonstrated in [11] to be the dominant strategy among those known in the public domain: outperforming not only all other public-domain algo trading strategies, but also out-performing human traders. However, [34] questions that claim, and recently BSE has been used as the test-bed for sequences of millions of simulated market sessions, discussed briefly in Section IX below, which cast further doubt on AA's dominance.

A notable omission from this list is the trading strategy first published by [15], and known as *GD*. This, operating in a modified form (*Modified-GD* or simply *MGD*), was demonstrated in 2001 by a team of IBM researchers [10] to be one of two automated trading algorithms that could consistently outperform human traders in a series of controlled laboratory experiments; the other trading algorithm that outperformed human traders in the IBM study was ZIP. MGD was subsequently [31] extended into a strategy known as GDX. GD/MGD/GDX are not included in the standard distribution of BSE purely so that students using BSE as a learning environment can be given the exercise of writing their own GD-derived traders.

BSE also includes samples of two extremely simple trading strategies, added purely as introductory illustration of how trading agents can be written to interact with BSE. These are: *Giveaway* (GVWY), a robot trader that ignores the LOB and instead immediately sets its quote price to exactly the same as its given limit-price, thereby surrendering any hope of making a profit on the trade but nevertheless maximizing its chances of finding a counterparty; and *Shaver* (SHVR) which reads data from the LOB and immediately sets its quote price to one penny (i.e. 0.01, the smallest unit of currency) less/more than the current best ask/bid when working a sell/buy order, stopping when its current quote is at the order's limit price.

VII. Supply and Demand Schedules

To fully specify a market experiment in BSE, as with any study in experimental economics, it is necessary to specify the nature of the supply and demand curves in the experiment. High-school microeconomics tells us that the intersection of the supply and demand curves is the equilibrium point, the ordinates of which are the theoretical equilibrium price P_0 and equilibrium quantity Q_0 for the market at that time. Studies in experimental economics typically concentrate on the speed and extent to which the transaction prices in the market converge on the underlying P_0 value, and the degree of variation in transaction prices measured relative to P_0.

In classic works from the experimental economics literature such as Smith's original paper on his first ever experiments [28] or Gode and Sunder's landmark zero-intelligence-trader paper [16], the shape and nature of the supply and demand curves, i.e. the *supply and demand schedule* (SDS), were decided *a priori* as part of the experiment design, and typically remained fixed for the duration of each experiment. If there was any change in the SDS, it was typically a single "market shock" where partway through an experiment the SDS that had been in play since the start of the experiment was replaced by a second SDS, one that differed from the first, which then remained in play until the end of the experiment. This allowed the response of the market to the shock-change in SDS to be recorded and analyzed. Such shock-change SDSs were also used to evaluate the response of algo trading strategies such as ZIP [8] and AA [35,36].

In BSE it is possible to specify a single SDS that remains static for the duration of an experiment, or to specify one or more shock-changes, jumping from one SDS to another, in the duration of an experiment. However, because real markets tend very rarely (arguably never) to have a static equilibrium price, and because sudden step changes in supply and/or demand do happen but are typically exceptional events, BSE also allows for the SDS to vary continuously over time. In the simplest case, the experimenter using BSE can specify an equilibrium offset function *EO(t)* that specifies a value at time *t* that is thereafter added to limit prices on trader assignments shifting the supply and demand curves up or down equally, without changing any other aspect of the SDS. Figure 2 illustrates a case in which the *EO(t)* function is periodic, defined by a simple periodic mathematical function.

BSE allows *EO(t)* functions to be defined as closed-form mathematical functions as illustrated in Figure 2, or as look-up-tables (LUTs). Using a LUT for *EO(t)* makes it possible to have the value of P_0 in an experiment be driven from a time-series of real market-data, such as intraday transaction prices or midprices for a specific equity or currency-pair. In such a use-case, the BSE underlying equilibrium price will vary over time in the same way as the real tradeable security, but the actions of the traders within the BSE experimental market can still have effects on the market's supply and demand, and hence on the subsequent price dynamics in the market.

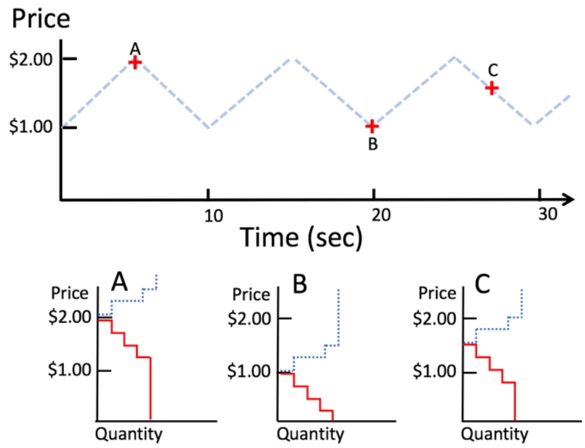

Figure 2: Continuously-varying supply-and-demand-schedules (SDSs). The upper graph shows an example continuously varying equilibrium offset function, where the equilibrium offset price is a function of time *t*: here the offset function is periodic, a triangle wave with a frequency of 0.1Hz, illustrated by the dashed blue line. Three points on the line have been highlighted; *A* at time *t*=5.0sec; *B* at *t*=20sec; and *C* at *t*=27.5sec. The lower three graphs show the supply and demand schedule in the market at points *A*, *B*, and *C* respectively: the underlying theoretical equilibrium price P_0 (to which transaction prices are expected to converge) is $2.00 at *A*, $1.00 at *B*, and $1.50 at *C*, but other than these vertical shifts caused by changes in the offset value, the shapes of the supply and demand curves remain unchanged.

BSE's specification of SDSs also allows for degrees of random jitter to be added the prices in the SDS; and for the arrival times of new assignments to either be simultaneously replenished for all traders (as is common in past experimental economics studies such as [28, 16, 8, 35, 36]), or to arrive as a nondeterministic stream, with the BSE experimenter having the ability to choose/define the stochastic process that governs arrival times of new assignments. Both the random jitter and the assignment arrival-times are under full control of the experimenter; complete details of specifying SDSs in BSE are given in the BSE documentation on *GitHub*.[1]

Figure 3 illustrates a time-series of transaction prices in a BSE market populated by 40 traders working buy orders and 40 traders working sell orders, with the flow of client orders specifically structured to give a P_0 that varies sinusoidally over time. The theoretical value of P_0 is shown by the solid line, and it can be seen that the transaction prices of the traders in the market are closely following the underlying P_0, with some lag determined by the time it takes for interactions among the traders engaged in price-discovery to reflect the change in P_0.

VIII. Using BSE in University Teaching

Since its first release as open-source on *GitHub* in 2012, BSE has been used successfully as the platform for illustrating various issues in trading technology in taught modules/units on MSc and MEng degrees at the University of Bristol. The source-code of BSE serves as an illustrative reference implementation showing how the core mechanism in a LOB-based exchange operates, and the various algo-trader strategies pre-coded into BSE also offer reference implementations so that students can study how descriptions of trading algorithms are translate into working program code: the lack of such reference implementations in the original source literature has been highlighted in [33] as a major obstacle to subsequent replication studies.

To keep things simple when students are first learning to use BSE, limits can be imposed so that all quotes have a fixed quantity of size 1, and so that each trader can only have one quote on the LOB: if a trader already has a quote on the LOB then the next time that trader issues a quote, its new quote replaces the old one. This simplified style of LOB in BSE is consistent with the stripped-down minimalism of Smith's initial pioneering studies in experimental economics [28] where similar simplifying constraints were introduced, establishing a tradition that has continued to present-day practice in experimental economics research (see e.g. [20, 21, 9]). The BSE platform has been used as the basis for coursework assessments where students are required to write their own trading algorithms, embed those algorithms into a trading agent on BSE, and empirically evaluate their algorithmic trader via an appropriately structured series of experiment and trials: this motivates the students to also learn generic skills in the design of experiments and in the statistical analysis of noisy empirical data.

A number of our masters-level (MEng and MSc) students have also used BSE as the basis for the major project that each student independently works on and documents in their individual master's thesis; a recent one of these has resulted in a publication in an international peer-reviewed academic conference [5]. The discussion here will focus on past use of BSE in teaching masters-level computer science students; however we intend, in early 2019, to roll out the use of BSE in new teaching content currently being developed, on the use of advanced data analytics in economics, finance, and management.

BSE was developed to give students experience of experimenting with automated algo-trading systems active in a market where those algo traders could themselves individually affect, and collectively determine, the trajectory of transaction prices over time. This is not possible when working with traditional financial-market simulators that simply regurgitate historical time series of transaction prices from a "tape", a database of asset prices over time. In a real market, if a large number of traders decide to sell their holdings of a specific asset (or, equivalently, if a single trader decides to sell a very large quantity of a single asset) then, all other things being equal, the increased supply of that asset into the market will depress the price. Traders working sufficiently large order-sizes need to be mindful of this because sometimes, merely revealing their intention to buy or sell a large quantity of an asset can result in the price that trader is offered being different from the best market price currently shown on the LOB or the time-series of transaction prices, as potential counterparties change their view of what the a fair price should be now, in anticipation of the change in price that will result once the large deal goes through. This effect is known as *market impact*: very many automated trading systems have to deal with market impact. Manifestly, market impact cannot be experienced when working solely from historical data tapes: the price of the asset at the time immediately after execution of a large trade in that asset will be whatever historical price is written on the tape, regardless of the quantity just traded whether it is for a quantity of one, or one million.

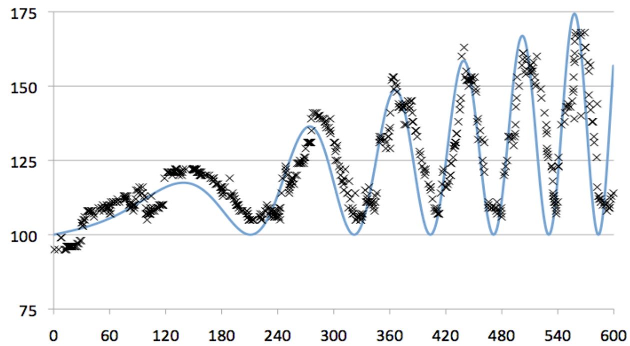

Figure 3: transaction price time-series from a BSE market where the constant stochastic arrival of client orders into the market is deliberately structured to give supply and demand schedules where the theoretical equilibrium price varies over time, following a sinusoidal path that grows in amplitude, indicated by the solid blue line. Horizontal axis is time in seconds; vertical axis is price. This market is populated with 40 buyers and 40 sellers, each group divided into 10 GVWY traders, 10 SHVR, 10 ZIC, and 10 ZIP; the crosses show individual transaction prices. See text for further discussion.

Over the years 2012-15 BSE was used at the University of Bristol as the basis for a major coursework assignment in our *Internet Economics and Financial Technology* unit: the unit was at that time assessed 50% by examination, and 50% by programming coursework where the students were required to submit a report on the design and implementation of a new trading agent for BSE. If students could not or did not want to think up their own trading agents, they were given the option of implementing the GD algorithm [15] which had deliberately not been implemented in BSE to leave open the opportunity of setting as an exercise for our students the task of writing an implementation of MGD. In some years we

[1] BSE *GitHub* site is https://github.com/davecliff/BristolStockExchange.

required the students' trading agents to be sales traders, as defined above; in other years we required the trading agents to work as *proprietary traders* (usually referred to as a "prop traders" within the industry), i.e. to start with a pot of money, and to then buy and sell assets in the market with the intention of turning a profit on the sequence of transactions, such that, at the end of a market session, the automated prop trader could sell off its holdings of assets, add the cash resulting to the sale to any other cash still held in its pot, and hope to end up with more cash than it started with. The primary difference between a prop trader and a sales trader is that a sales trader works client orders that are delivered to it from an external source, whereas a prop trader is responsible for generating its own endogenous trading signals, deciding when to buy and when to sell; this makes it somewhat more challenging to arrive at a working algo strategy for prop traders.

Over the years that we based the programming project on BSE a total of around 250 students used it, and it met with enthusiastic responses in end-of-module feedback (with one notable exception, a single unhappy student who described BSE in very negative terms; just going to prove that it is very difficult to do anything that keeps *everyone* happy). Many of our Bristol students whose first experience of trading in financial markets was gained working with BSE are now enjoying careers working for major investment banks and fund-management companies.

IX. Using BSE in Research

BSE has served as the platform on which several of our masters-level students have based the work that they undertook for their final master's thesis. Recent Bristol masters-theses [32, 4] have explored the use of "deep learning" neural networks (DLNNs: see e.g. [23, 25]) to investigate whether deep learning can be used to replicate the behavior of an adaptive trader in a market purely from observation of that trader's actions in the market, and the client-orders that it is working. Promising early results with DLNN successfully replicating ZIP traders for live trading in BSE are reported in [5], with subsequent extensions and replications of this work in [17, 18]. Of particular note is [18] which successfully demonstrated the use of automated design/optimization techniques to create the DLNN architecture: in prior work [32, 4], the DLNN architecture (i.e., the number of layers in the network, and the number of units in each layer) was optimized by hand, in a process that was essentially trial-and-error with some educated guesswork; [18] showed that high-performing network architectures could be automatically discovered by standard machine-learning and optimization techniques, such as genetic algorithms.

Most recently, [9] reports the use of BSE as the platform for running many hundreds of thousands of independent market experiments, exploring a result first reported in 2015 by Vach [34]. Vach reported that whether Vytelingum's AA strategy dominates GDX in an experiment is highly dependent on the way the experiment is structured, on the relative proportions of AA and other strategies in the experiment's population of traders: put simply, AA had previously been identified in [35, 11] as the strategy that dominates all others, but Vach demonstrated that in fact whether GDX is dominated by AA or not depends on the specific proportion of the two trading strategies in the experiment. Sometimes AA wins; other times GDX wins. In [9], experiments are reported that vary the proportions of four trading strategies: AA, SHVR, ZIC, and ZIP, working with an equal number T of traders (buyers and sellers) in each experiment, varying the total population sizes $P=2T$ from T=4 to 16, and systematically varying the ratio of AA:SHVR:ZIC:ZIP from 0:0:0:P, through all possible ratio combinations to the situation where the four strategies are represented equally with ratio (P/4):(P/4):(P/4):(P/4), and then on through all other combinations, ending at the ratio P:0:0:0. The combinatorics of such an experiment are quite explosive, and the need to generate statistically rigorous results mean that at any one ratio, for any one T, it is desirable to perform a large number N of independently repeated trials: N=100 is a safely high value, but this means that any one experiment, working through all ratios for each T, and working through a reasonable range of T values, can require hundreds of thousands of trials (i.e., individual market sessions) to be performed for any one SDS. Varying across a reasonable number of qualitatively different SDSs can take the total number of trials required, the total number of independent market sessions simulated on BSE for one experiment, beyond one million. Thankfully, with the use of cloud computing services now commonplace, such large numbers of trials can be distributed over multiple (virtual) machines, because each trial is statistically independent from all the others. The results presented in [9] confirm Vach's findings: whether AA dominates other strategies or not does indeed seem to be heavily reliant on the specific ratios of the different strategies present in the market, and on the nature of the SDS used in the market experiment. And, for this reason, AA can no longer be described as the strategy that dominates all others. See [9] for further details.

It is perhaps no surprise that when running hundreds of thousands of BSE market trials, the size of the data files that result are substantial: many gigabytes. It's worth noting that, in principle, public repositories like *GitHub* can be used not only to publish open-source code, but also to make large data-files open for inspection, validation, and re-analysis by other researchers. In this way, BSE offer the opportunity to serve as a common public platform for running experiments that study automated trading technology and market microstructure.

The list of order types given in Section III will come as no surprise to anyone familiar with the development of electronic markets over the past 20 years, but it is worth noting that much of the AI/Agents research on trading strategies seems stuck in the methodological mind-set of Smith's initial experiments from more than half a century ago, developing trading agents that are limited to issuing MKT and LIM orders; a point explored in more depth in [9].

X. Conclusion

BSE was initially designed for teaching masters-level computer science students about financial technology, specifically automated trading systems and the internal workings of contemporary financial exchanges. In the six years since it was first made available on *GitHub* it has proven to be a useful resource, and many University of Bristol graduates whose first encounter with fintech was via BSE have since gone on to successful careers in major investment banks and fund-management companies. BSE's level of realism is sufficiently high that it can also be used as a platform for research experiments. Coupled with the ready availability of elastically scalable cloud-computing resources, it is now perfectly feasible to run many hundreds of thousands of market experiments in only a few hours, and hence to explore wide volumes of parameter space. Results from experiments run on BSE can be readily checked and independently repeated/replicated by other researchers around the world, with BSE thereby having the potential to become a common public-domain platform not only for teaching but also for research exploring issues in automated trading and market microstructure.

Acknowledgments

Thanks are due to Ash Booth for uploading his AA trader Python code to the BSE *GitHub* repository; and to Natalie Lang for her forensic analysis of the BSE source-code in early 2018, which identified some improvements. Many thanks to the people who via *GitHub* have provided pull requests, bug reports, forks, and revisions. Thanks also to the approximately 250 masters-level students at the University of Bristol who have used BSE and provided feedback on it since 2012.

References

[1] M. Aquilina, M. Eyles, J. Shao, and C. Ysusi. *How do participants behave during flash events? Evidence from the UK equity market.* Research Note, UK Financial Conduct Authority, 2018.

[2] S. Arnak and J. Saluzzi, *Broken Markets*. Prentice Hall, 2012.

[3] H. Bodek and S. Dolgopolov *The Market Structure Crisis: Electronic Stock Markets, HFT, & Dark Pools.* Decimus, 2015.

[4] A. le Calvez, *Learning to be a Financial Trader: An exploration of Neural Networks in a Continuous Double Auction*. MSc Thesis, Department of Computer Science, University of Bristol, 2018.

[5] A. le Calvez and D. Cliff. "Deep Learning can Replicate Adaptive Traders in a Limit-Order-Book Financial Market". To appear in *Proceedings of the IEEE Symposium on Computational Intelligence for Financial Engineering (CIFEr2018).* In press, 2018.

[6] Á. Cartea, S. Jaimungal, and J. Penalva, *Algorithmic and High-Frequency Trading.* Cambridge University Press, 2015.

[7] J. Cartlidge and Cliff "Exploring the 'robot phase transition' in experimental human-algorithmic markets" UK Government Office for Science, Foresight Driver Review DR25, https://bit.ly/2llHjbh, 2012.

[8] D. Cliff, "Minimal-Intelligence Agents for Bargaining Behaviours in Market-Based Environments". Hewlett-Packard Labs Technical Report HPL-97-91, 1997.

[9] D. Cliff, "Borderline-stupid traders beat advanced adaptive AI systems in realistically dynamic continuous double auction markets" Unpublished, submitted for review, 2018.

[10] R. Das, J. Hanson, J. Kephart, and G. Tesauro, "Agent-Human Interactions in the Continuous Double Auction." In *Proceedings of the International Joint Conference on Artificial Intelligence* (IJCAI), 2001.

[11] M. De Luca, and D. Cliff, "Human-Agent Auction Interactions Adaptive-Aggressive Agents Dominate." In *Proceedings of the International Joint Conference on Artificial Intelligence* (IJCAI), 2011.

[12] M. De Luca, J. Cartlidge, C. Szostek, and D. Cliff "Studies of interactions between human traders and algorithmic trading systems" UK Government Office for Science, Foresight Driver Review DR13. https://bit.ly/2llv52c, 2012.

[13] M. De Luca, *Adaptive Algorithmic Trading Systems.* PhD thesis, Department of Computer Science, University of Bristol, 2016.

[14] M. Duffin and J. Cartlidge, "Agent-based model exploration of latncy arbitrage in fragmented financial markets". To appear in *Proceedings of the IEEE Symposium on Computational Intelligence for Financial Engineering (CIFEr2018).* In press, 2018.

[15] S. Gjerstad and J. Dickhaut, "Price Formation in Double Auctions", *Games and Economic Behavior,* 22(1):1-29. 1998.

[16] D. Gode and S. Sunder, S. "Allocative efficiency of markets with zero-intelligence traders: Market as a partial substitute for individual rationality", *Journal of Political Economy*, 101(1):119-137, 1993.

[17] B. Hackett, *Using Deep Learning to Replicate Sales-Trader Behaviour in a Limit Order Book Financial Exchange.* MSc Thesis, Department of Computer Science Department, University of Bristol, 2018.

[18] C. Higgins, *Learning to Trade: Using Deep Recurrent Neural Networks to Emulate a Trading Strategy in a Simulated Financial Exchange.* MSc Thesis, Dept Computer Science, Uni. of Bristol, 2018

[19] N. Johnson, "To slow or not? Challenges in sub-second networks", *Science* 355:801, 2017.

[20] J. Kagel and A. Roth, *The Handbook of Experimental Economics, Volume 1,* Princeton Universty Press, 1997.

[21] J. Kagel and A. Roth, *The Handbook of Experimental Economics, Volume 2*, Princeton Universty Press, 2016.

[22] P. Kovac, *Flash Boys: Not So Fast*. Directissima Press, 2014.

[23] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet classification with deep convolutional neural networks". In *Proc. 25th International Conference on Neural Information Processing Systems (NIPS'12)*, F. Pereira, F, *et al.* (Eds.), Curran Associates, pp.1097-1105, 2012.

[24] M. Lewis, *Flash Boys: Cracking the Money Code.* Penguin, 2014.

[25] J. Patterson and A. Gibson, *Deep Learning: A Practitioner's Approach.* O'Reilly, 2016.

[26] S. Patterson, *Dark Pools: The Rise of AI Trading Machines and the Looming Threat to Wall Street.* Cornerstone, 2012.

[27] J. Rust, J. Miller, and R. Palmer, R. "Behaviour of trading automata in a computerized double auction market" in Friedman, D., Rust, J. (Eds) *The Double Auction Market: Institutions, Theories, & Evidence.* Addison Wesley, pp.155-198, 1992.

[28] V. Smith, "An experimental study of competitive market behavior" *Journal of Political Economy*, 70(2):111-137, 1962

[29] S. Stotter, J. Cartlidge, and D. Cliff, "Exploring assignment-adaptive (ASAD) trading agents in financial market experiments," in *Proc. 5th Int. Conf on Agents & Art. Intell.*, J. Filipe & A. Fred, Eds. SciTePress, Vol1, pp.77-88, 2013.

[30] S. Stotter, J. Cartlidge, and D. Cliff, "Behavioural investigations of financial trading agents using Exchange Portal (ExPo)." In: N. Nguyen *et al.,* Eds, *Trans on Computational Collective Intelligence XVII.* Springer, pp. 22-45, 2014.

[31] G. Tesauro and J. Bredin, "Sequential Strategic Bidding in Auctions using Dynamic Programming". In *Proc. First Int. Conference on Autonomous Agents and Multi-Agent Systems* (AAMAS), 2002.

[32] K. Tibrewal, *Can Neural Networks be Traders? Explorations of Machine Learning Using the Bristol Stock Exchange*. MEng Thesis, Department of Computer Science, University of Bristol, 2017.

[33] I. Toft, *Adaptive Agents for Single and Double Auctions*. PhD Thesis, Department of Computer Science, University of East Anglia, 2007.

[34] D. Vach, *Comparison of double auction bidding strategies for automated trading agents.* Masters Thesis, Institute of Economic Studies, Charles University in Prague, 2015.

[35] P. Vytelingum, *The Structure and Behaviour of the Continuous Double Auction*. PhD Thesis, School of Electronics and Computer Science, University of Southampton, 2006.

[36] P. Vytelingum, D. Cliff, and N. Jennings, "Strategic Bidding in Continuous Double Auctions". *Artificial Intelligence*, 172(14):1700-1729, 2008.

[37] E. Wah and M. Wellman. "Latency arbitrage in fragmented markets: A strategic agent-nased analysis," *Algorithmic Finance*, 5(3-4):69-93, 2016.